Remarks

As stated above, Applicants appreciate the Examiner's thorough examination of the subject application and request reexamination and reconsideration of the subject application in view of the preceding amendments and the following remarks.

As of the office action of February 18, 2009, claims 1-17 were pending in the subject application, of which claim 1 is an independent claim. With this response applicants have amended claim 1 and cancelled claims 10, 12, and 13.

A. 35 U.S.C. § 103 Rejections

The examiner rejects claims 1, 2, 5-7, and 9-16 under 35 U.S.C. § 103(a) over U.S. Patent Application Publication 2004/0167896 ("Eakin") in view of U.S. Patent 6,901,595 ("Mukundan"). *Office Action page 2.* Applicants respectfully traverse this rejection.

In an effort to advance prosecution, Applicants have amended independent claim 1. Applicants' newly amended independent claim 1 is provided below for the Examiner's convenience.

> 1. (Currently Amended) A computer program residing on a computer readable medium having a plurality of instructions, which, when executed by a processor, cause the processor to perform operations comprising:
> connecting a portal to one or more user interface (UI) components;
> linking one or more interface (UI) components to a repository layer and connectivity layer through an object access layer, the repository layer including metadata pertaining to roles, work sets and personalization information, the metadata configured to interact with at least one template, the at least one template providing a format of information according to preset conditions, the at least one template configured to interact with Web application server (WAS) processes and core restructuring processes;
> linking the repository layer and the connectivity layer to source systems;
> accessing a database that includes data representing multiple enterprise functions, wherein the data representing multiple enterprise functions includes personal tasks and resources for users; and

> using one or more object modeling tools, one or more process modeling tool, and the one or more UI component to build components of cross-functional applications from the data representing multiple enterprise functions, wherein the cross-functional applications include pages that display the personal tasks and resources for users. *Applicants' claim 1. Emphasis Added.*

With this amendment, Applicants have incorporated the subject matter of dependent claims 10, 12, and 13 into independent claim 1. Support for this amendment may be found throughout the subject application, for example, in paragraph [0052], which has been provided below for the Examiner's convenience.

> [0052] The databases and repositories in the persistence/repository layer 610 can contain metadata. Metadata refers to data that describes other data, such as data pertaining to roles, work sets and personalization information, for example. The metadata can interact with the object access layer 608, connectivity layer 612 and application services logic 606. The metadata can also interact with templates 616. The templates 616 provide a format or organization of information according to preset conditions. The templates 616 can interface with Web application server (WAS) processes 618 and core merger processes 620 in the repository layer 610. *Subject application, para. [0052].*

Applicants respectfully submit that neither Eakin nor Mukundan disclose each and every limitation of Applicants' newly amended claim 1. Specifically, Applicants contend that neither of these references disclose "linking one or more interface (UI) components to a repository layer and connectivity layer through an object access layer, the repository layer including data pertaining to roles, work sets and personalization information, the metadata configured to interact with at least one template, the at least one template providing a format of information according to preset conditions, the at least one template configured to interact with Web application server (WAS) processes and core restructuring processes."

Referring to page 5 of the Official Action, the Examiner seems to suggest that Eakin discloses "metadata comprises data pertaining to roles, work sets and personalization information (Eakin; ¶[0052-0053] and [0068])." These passages of the Eakin reference have been provided below for the Examiner's convenience.

> [0052] The deployed metadata set 334 consists of a description of the relationships between the properties in the set, the native type binding of those properties, and the binding to the storage layer. The relationships between the properties in a metadata set 334 can be described in a general way so that the metadata description can be deployed to different containers that may be implemented on different data platforms. The native type binding description as defined by resource description framework (RDF) mapper 362, RDF language binder 364, and RDF storage binder 366 is specific to a programming language and is used to generate code (e.g., in code generator 350) that implements the binding. RDF storage binder 366 allows properties in multiple metadata sets to map to a single value in storage (the canonical property). Part of the storage binding defines the transcoding required in transcoder 320 to transform a property value encoded in storage driver 310 into the proper encoding for a specific metadata set 334.
>
> [0053] A client application 130, such as content portal 134, makes calls on the metadata store 128 via metadata set interface 332 to the object that holds the values of the properties, and on any metadata sets 334 integrated in the metadata framework. A storage driver 310 manages the persistence of property values. FIG. 3 further illustrates the flow of data between the components of the metadata store 128. The storage driver 310 provides native type binding through a generic, Java database connectivity (JDBC)-like API or another suitable API. The higher-level metadata set API delivers property values not only in the proper native type but also in the proper encoding for that metadata set. An application can choose to use such an API in order to take advantage of the metadata transcoding facilities built into the metadata store 128 and to avoid having metadata mappings for each component in an application.
>
> [0068] In some embodiments, the content portal 134 is configured to support an operator role of a digital asset reviewer. In these embodiments (not depicted in the flow diagram of FIG. 5), a digital asset reviewer accesses the metadata associated with a digital asset 110 of interest. The reviewer, after accessing and observing or otherwise verifying the quality or accuracy of the human-readable information therein is given the opportunity to modify a metadata parameter to indicate approval or disapproval of the present state of the digital asset 110. In either case, the digital asset reviewer

> entered metadata parameter is further associated with the digital asset 110 and an indication thereof can be observed by operators serving other roles. *Eakin, paras. [0052-0053] and [0068].*

Applicants are unable to find reference to a "repository layer including metadata pertaining to roles, work sets and personalization information" in this or any of the other passages of the Eakin reference. It is Applicants' understanding that these passages mention the terms "roles" and "metadata", however, Applicants respectfully submit that the limitation as a whole is not taught by these passages.

Further, the Examiner also seems to indicate that Mukundan discloses "metadata interacts with the templates, the templates providing a format of information according to preset conditions (e.g., execute a RPC; Mukundan, Figure 11, col. 15: lines 1-30)." This Figure and the accompanying passages from Mukundan have also been provided below for the Examiner's convenience.

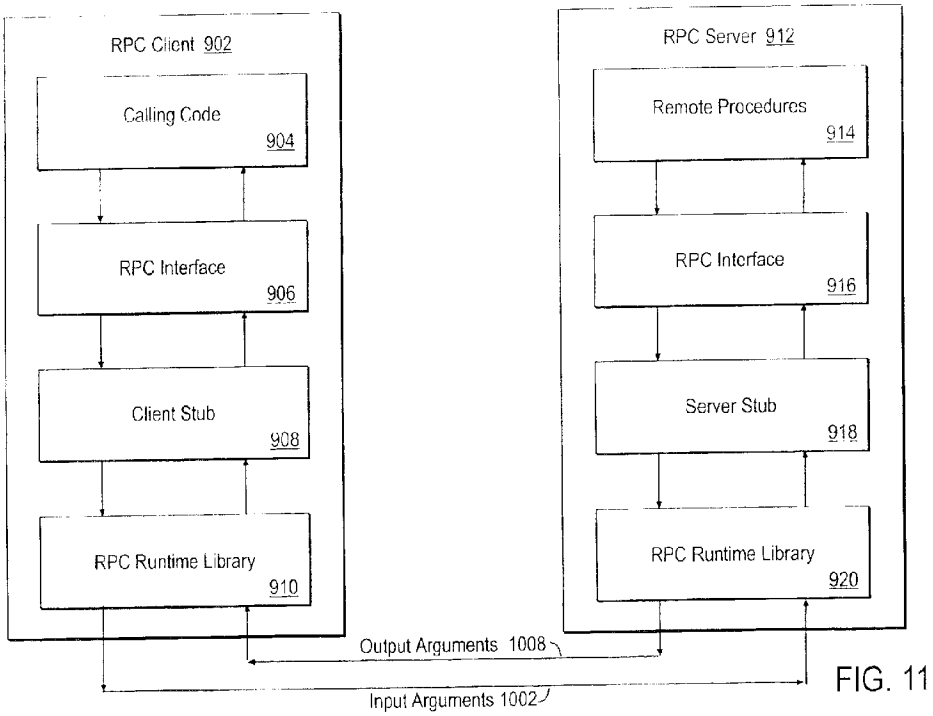

FIG. 11

> FIG. 11 generally shows exemplary roles of RPC application code segments 904 and 914, RPC interfaces 906 and 916, RPC stubs 908 and 918, and RPC runtime libraries 910 and 920 during a remote procedure call. The client's application code or calling code 908 invokes a remote procedure call, passing the input arguments 1002 through the client's RPC interface 906 to the client stub 908. The client stub 908 marshalls the input arguments 1002 and dispatches the call to the client's RPC runtime library 910. The client's RPC runtime library 910 transmits the input arguments 1002 to the server's RPC runtime library 920, which dispatches the call to the server stub 918 for the RPC interface 916 of the called procedure. The server's stub 918 unmarshalls the input arguments 1002 and passes them to the called remote procedure 914. The server's application code or remote procedure 914 executes and then returns any output arguments 1008 to the server stub 918. The server stub 918 marshalls the output arguments 1008 and returns them to the server's RPC runtime library 920. The server's RPC runtime library 920 transmits the output arguments 1008 to the client's RPC runtime library 910, which dispatches them to the client stub 901. The client's stub 908 unmarshalls output arguments 1008 and returns them to the calling code 904. *Mukundan, col. 15, lines 1-30.*

Applicants are unable to find reference to "the metadata configured to interact with at least one template, the at least one template providing a format of information according to preset conditions" in this or any other section of Mukundan. Specifically, Applicants are unable to locate any discussion of "metadata", "templates" or "preset conditions" in this passage. Applicants do not understand how Mukundan, without disclosing these elements, could possibly disclose this limitation when viewed as a whole.

Moreover, the Examiner also seems to indicate that Mukundan discloses "the templates interact with Web application server (WAS) processes and core restructuring processes (Mukundan; col. 19: lines 18-31)." This passage has also been provided below for the Examiner's convenience.

> FIG. 15 shows an exemplary process 1500 of communication in which the browser-side applet invokes the method directly on the JSSBusComp object. It should be noted that the process 1500 of FIG. 15 will be described below as though the process had occurred in a multiple-device configuration shown in FIG. 6A. In block 1505, the browser-side applet, JSSApplet, redirects the method to the JSSBusComp object. In block 1510, the client-side business component object, JSSBusComp, would then issue a remote procedure call, through JSSApplication object, to a server-side business component, CSSBusComp. The server-side business component generally processes the RPC method call, sets and sends back a status, and also returns a set of notifications in appropriate cases (block 1515). *Mukundan, col. 19, lines 18-31.*

Applicants are unable to find reference to "the at least one template configured to interact with Web application server (WAS) processes and core restructuring processes" in this or any other passage of Mukundan. Applicants respectfully request further clarification regarding this rejection.

In light of the above, Applicants respectfully submit that newly amended independent claim 1 is in condition for allowance. Since the remaining dependent claims depend, either directly or indirectly, from Applicants' newly amended claim 1, Applicants respectfully request that these claims are in condition for allowance as well.

In consideration of the amendments and foregoing discussion, the application is now believed to be in condition for allowance. Early allowance of the subject application is respectfully solicited.

This response is not believed to necessitate any additional fees. However, in the event that additional fees are due, please charge or credit any refund to our Deposit Account No. 50-2324.

Respectfully Submitted,

Dated: May 18, 2009

/Brian J Colandreo/
Brian J Colandreo
Reg. No. 42,427

Holland & Knight LLP
10 St. James Avenue
Boston, MA 02116-3889
Telephone 617-305-2143
Facsimile 617-523-6850

6136466_v1